Vision-Based Autonomous Vehicle Navigation in Complicated Room Environments with Collision Avoidance Capability by Simple Learning and Fuzzy Guidance Techniques

Yi-Chieh Chen and Wen-Hsiang Tsai
Department of Computer and Information Science
National Chiao Tung University
Hsinchu, Taiwan 300

***Abstract* — A vision-based fuzzy approach to autonomous vehicle navigation in complicated room environments with obstacle avoidance capability is proposed. In the learning stage, through a simple non-visual strategy by driving a vehicle around a multi-room environment freely, an inter-room path map is constructed by analyzing the recorded user-driving control information and the odometer data. And following the learned map in the form of an attributed undirected graph, the vehicle can accomplish inter-room navigation and collision avoidance by several proposed fuzzy guidance techniques using environment parameters extracted from along-path images acquired by an on-board vision system. And measures for navigation accuracy maintenance are also proposed. Good experimental results show the feasibility of the proposed approach.**

1. INTRODUCTION

Recently, vision-based autonomous vehicles or mobile robots have been used in more and more human environments, especially in security patrolling and home service applications. A difficulty encountered in such vehicle navigation applications is the complicated environment faced by an autonomous vehicle, resulting in a challenge of designing a general and flexible learning strategy for various navigation environments. Most former works [1-5] focused on *vision-based* learning. Though this approach is useful for constructing visual environment data, which can then be used for locating the vehicle in the navigation stage, yet certain restrictions are usually imposed on the learning process, resulting in inconvenience or difficulty in conducting the learning work. Furthermore, certain artificial landmarks or specific scene features are often forced to appear in the environment to be learned, in order to accomplish the work of locating the vehicle by landmark or feature matching in the navigation stage. This often makes the learning method inapplicable or less flexible in certain applications. Finally, use of landmark or feature matching often yields imprecise vehicle location results, leading possibly to unstable navigation performances. In this paper, we propose a *non-visual* approach to learning which solves all the above-mentioned problems encountered in the conventional vision-based learning approach.

The proposed non-visual learning process is just a sequence of *free* actions of vehicle driving through a number of reachable rooms without using a vision system. The vision system is used in the navigation stage mainly for the purpose of collision avoidance. Only simple data consisting of the records of user-driving actions as well as the odometer values of traversed paths are collected, followed by the execution of a path map creation process with the learned data as input. The generated path map consists of a set of connected nodes, with each inter-node edge being a traversed trajectory in the learning process.

In the navigation stage, the main task is to guide the vehicle to follow a selected path generated from the learned path map. The path information consists of a set of driving actions and odometer values. A *vision-based* navigation strategy based on *fuzzy-set theory* is proposed. The reason why we adopt the fuzzy-control approach is three-fold. First, general indoor environments, especially those in rooms with furniture and decorations, are quite complicated, and conventional vision-based methods requiring precise image analysis for vehicle location computation are mostly inapplicable. Second, the adopted fuzzy-control approach on the contrary allows the vehicle system to infer navigable space along the path in a less accurate way. And lastly, fuzzy computation, when combined with our collision-avoiding strategy of vehicle guidance, becomes simple and effective, as experienced from our experiments. The proposed collision-avoiding guidance strategy is based on the use of a computer vision system using a wireless camera, and regards objects along navigation routes as obstacles, which should be avoided. In essence, we keep equilibrium between navigation accuracy and fuzzy control.

In the remainder of this paper, the proposed vehicle system configuration and the overall navigation procedure is sketched in Section 2. In Section 3, the proposed learning process is described. In Section 4, the proposed fuzzy guidance techniques are detailed. Some experimental results are shown in Section 5, followed by some concluding remarks in the last section.

2. SYSTEM CONFIGURATION AND NAVIGATION PROCEDURE

In this study, we use as our autonomous vehicle the Amigo Robot, a mini-vehicle made by ActivMedia Robotics Technologies, Inc. There are an odometer and a wireless camera on the vehicle. Through the wireless device

and the system software on the vehicle, we can control the vehicle with a remote PC. The max advance and rotation speeds of the robot are 75 cm/sec and 300 degrees/sec, respectively, and either of the two speed encoders on the vehicle encodes 39000 ticks per wheel revolution with 124 ticks per millimeter. Although there are ultrasonic sensors on the vehicle, they are not used in this study.

A description of the major steps of the proposed system processes, including learning and navigation, is as follows.

Stage 1: learning

1.1 *Free manual driving* --- In this major step, we drive the vehicle from one room spot to another as wished, recording all the driving actions and information, including turnings with specified directions, and backward and forward moves with traversed distances.

1.2 *Automatic path map creation* --- In this step, we identify two types of path nodes, *check node* and *turn node*, in the path data collected in the last step to create a path map of the visited room spots in the form of a graph with undirected edges specifying the navigated routes.

Stage 2: navigation

1.3 *Path generation* --- In this step, a user selects a starting spot and an ending one for navigation, and the vehicle system generates accordingly a path consisting of a sequence of nodes and edges, with each node including its corresponding space coordinates and each edge including the labels of its two end nodes.

1.4 *Path traverse* --- The vehicle then starts to navigate along the generated path by visiting the nodes sequentially through the routes specified by the inter-node edges, using the fuzzy guidance techniques of line following between check nodes and curve following between turn nodes. Precision maintenance measures are also carried out at each node.

3. PROPOSED LEARNING PROCESS

As mentioned previously, the learning strategy proposed in this study is non-visual and uses no landmark or special features for vehicle location in the navigation environment. So, the vision system on the vehicle is disabled during the learning process.

3.1 Free Manual Driving Step

During the previously-mentioned manual driving step in the learning stage, the proposed system allows a user to control the vehicle to move freely in an unknown environment through a simple control interface. Each action taken by the user is recorded as a command together with its execution time with respect to the start time of the learning process. And the location of the vehicle, including its position and direction, is recorded every fixed time period (say, every second). A list of the labels for data representing the user-driving actions and information is shown in Table 1. All the recorded data are called *learned data* in the sequel. Each command is associated with three or four parameters. More specifically, either of the 'turn leftward' or the 'turn rightward' command is associated with the parameters of turn angle, position coordinates, direction angle, and action time; and each of the 'move forward', the 'move backward', and the 'stop' command with the parameters of position coordinates, direction angle, and action time (without the turn angle).

TABLE 1
USER-DRIVING COMMANDS AND PARAMETERS FOR LEARNED DATA

TYPES	COMMAND & PARAMETERS	LABEL OR VALUE
command	move forward	f
command	move backward	b
command	turn leftward	ℓ
command	turn rightward	r
command	stop	s
parameter	turn angle	θ_t
parameter	position coordinates	$(x{:}y)$
parameter	direction angle	θ_d
parameter	action time	t

3.2 Automatic Path Map Creation Process

The proposed automatic path map creation process, as mentioned previously, is designed to create a graph composed of two kinds of nodes, check node and turn node, from the learned data. An algorithm is proposed as follows for this purpose.

Algorithm 1. *Automatic path map creation process.*

Step 1. Take the learned data as input, and identify check nodes and turn nodes by processing the commands sequentially in the input in the following way. Let the currently processed command be denoted as C^0, the previously processed one as C^{-1}, and the next processed one as C^{+1}.

Step 1.1. If C^0 is 'move forward', then

a. if C^{-1} is 'turn leftward' or 'turn rightward,' then create a turn node at the spot visited by the vehicle three seconds after the action time of C^0 (i.e., three seconds after C^0 was executed);

b. if C^{+1} is executed at a spot farther than a meter from C^0, then create check nodes at spots with distance intervals of one meter with C^0 as the first created check node.

Step 1.2. If C^0 is 'stop', then

if the C^{+1} is 'turn leftward' or 'turn rightward', then create a turn node at a spot visited by the vehicle three seconds

before the action time of C^0 (i.e., three seconds before C^0 was executed); otherwise, create a check node at C^0.

Step 2. Repeat Step 1 until all commands in the input are processed.

In the above algorithm, the two threshold values of three seconds and one meter may be varied to meet other application needs. It can be seen from the algorithm that the resulting path map is a simple attributed undirected graph, which we found sufficient for this study.

4. Proposed Fuzzy Guidance for Navigation

In every navigation cycle, an input image captured with the wireless camera is processed to obtain two kinds of features, namely, *collision-free direction*, and *degrees of collisions of the left and the right route sides*. These features are taken as input into a fuzzy guidance algorithm, which yields an angle as output for use to steer the vehicle in each navigation cycle. In the following, we first describe how we compute the two kinds of features in Section 4.1, and then present the guidance algorithm in Section 4.2.

4.1 Computing Features for Use in Guidance

Image processing techniques are applied in this study to captured images to compute the features for use in vehicle guidance, as described in the following.

A. Computing collision-free direction

The feature of collision-free direction means the direction into which the vehicle may be driven with no collision with the objects along the path navigated by the vehicle. Such a feature is computed in every navigation cycle in the following way. First, we want to divide the input image into 4×4 blocks and classify each image block into two classes, namely, *route area* and *non-route one*. As a preliminary step to achieve this goal, we utilize an algorithm presented in Li and Tsai [6] to locate image blocks of possible route areas. The essence of the algorithm is to use two pixel features, namely, a pixel's gray-scale value and its Sobel edge value, to identify *candidate route-area pixel*. A pixel in the input image with its gray-scale value close to a pre-learned gray-scale value of the room ground and with its Sobel edge value smaller than a pre-selected threshold is classified as a candidate route-area pixel. An example of such image pixel classification results is shown in Figs. 2(a) and 2(b). Because of color and uniformity similarities, some wall or furniture regions may be misclassified as route areas, as can be seen in Fig. 2(b). Nevertheless, we propose an algorithm in this study to remove such erroneous areas in the following, which in addition computes the above-mentioned feature of collision-free direction from the remaining correct route areas in the input image.

Algorithm 2. *Computation of route areas and collision-free direction*.

Step 1. Perform region growing to find as the desired route area the largest bottom region in the candidate route-area pixels extracted from the input image using [6]. The region growing result of the above example is shown in Fig. 2(c).

Step 2. Put two parallel horizontal scanning lines in a fixed lower part of the route area. If any non-route area crosses either scanning line and cuts it into several line segments, pick out the longest segment and call it a *non-obstacle segment*; otherwise, the original scanning line is selected as the non-obstacle segment. At the end of this step, two non-obstacle segments will be obtained.

Step 3. Find the middle points of the two non-obstacle segments and connect them to form a line segment which we call *route segment*. The found middle points for the above example are A and B as shown in Fig. 2(c).

Step 4. Compute the middle point of the route segment, which we call *route center*. The result of this step for the last example is point C in Fig. 2(c).

Step 5. Connect the route center to the middle point of the bottom line of the image to form a line segment, which we call *guidance line*, like the line labeled L in Fig. 2(d).

Step 6. Find as the desired collision-free direction θ_g the angle of the guidance line with respect to the bottom line of the image.

In the above algorithm, no complicated 3D computer vision technique is used in computing the collision-free direction, as contrasted with conventional methods. Also, notice that by the above algorithm, the vehicle automatically has the capability of obstacle avoidance. Actually, obstacles in the path are treated similarly to along-path side objects (like furniture) in this study.

B. Computation of degrees of collisions

The degrees of collisions of the left and the right route sides are defined and computed in this section. We first define a rectangular window in each captured image with two sub-windows separated by a centerline, as shown in Fig. 2(e). Then the proportion of the route area in the left sub-window is defined to be the degree of collision of the left route side, which will be denoted by P_L. That of the right route side can be defined similarly, and denoted by P_R.

4.2 Proposed Fuzzy Guidance

After the features described in Section 4.1 are acquired, a *steering angle* θ_s^i is computed for use in turning the vehicle in the ith navigation cycle. This angle θ_s^i is an output of a fuzzy guidance algorithm proposed in this study and described in the next section. If θ_s^i is positive, it means that the vehicle should turn leftward for the angle of θ_s^i; if θ_s^i is negative, it means that the vehicle should turn rightward for θ_s^i. The proposed fuzzy guidance algorithm is based on two fuzzy rules in terms of three linguistic variables LESS,

LEFT, and RIGHT that describe the values of the input features. The rules are described as follows:

Rule 1: if the collision-free direction θ_g trends to LEFT and the degree of collision of the right route side P_R is LESS, then turn the vehicle rightward for the angle of θ_s^i;

Rule 2: if θ_g trends to RIGHT and P_L is LESS, then turn the vehicle leftward for θ_s^i.

To implement the above two rules, six membership functions $\mu_{LEFT}(\theta_g)$, $\mu_{RIGHT}(\theta_g)$, $\mu_{LESS-R}(P_R)$, $\mu_{LESS-L}(P_L)$, $\mu_{f1}(\theta_S{}^i)$, and $\mu_{f2}(\theta_S{}^i)$ are defined as shown in Fig. 3, in which the units of θ_g, and P_R and P_L are degree and percentage, respectively.

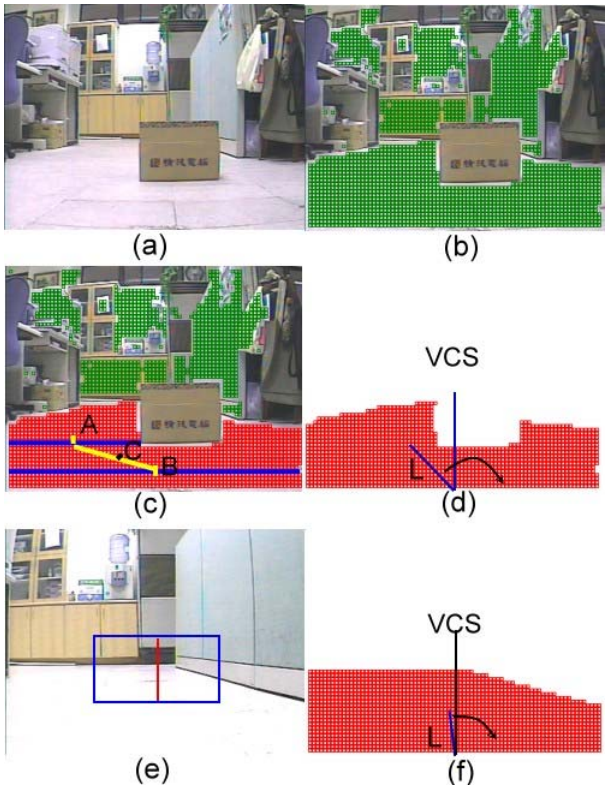

Figure 2. (a) An input image. (b) Result of candidate route point classification. (c) Result of route area extraction. (d) Result of guidance line computation. (e) Another input image. (f) Result of guidance line computation of (e).

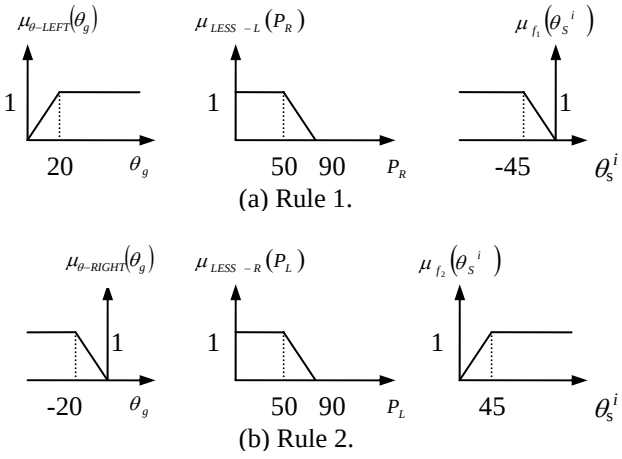

Figure 3. Membership functions

The fire strengths of Rules 1 and 2 can be calculated accordingly as follows:

$$m_1 = \mu_{LEFT}(\theta_g) \wedge \mu_{LESS-R}(P_R)$$

$$m_2 = \mu_{RIGHT}(\theta_g) \wedge \mu_{LESS-L}(P_L)$$

where $\wedge$ denotes the AND operator which is defined as the minimum function. After the fuzzification stage, the conclusion of each rule can be derived according to fuzzy reasoning. Because the membership function of the conclusion of each rule is monotonic, the Tsukamoto defuzzification method [7] can be applied to our fuzzy reasoning here. The crisp output θ_s^i, which is the desired steering angle, is calculated by the following equation:

$$\theta_S{}^i = \frac{\sum_{j=1}^{2} m_j \mu_{\theta_S{}^j}^{-1}(m_j)}{\sum_{j=1}^{2} m_j}$$

where m_k is the firing strength of Rule k, and $\mu_{\theta_S{}^j}^{-1}(m_j)$ is the inverse function of $\mu_{\theta_S{}^j}(\theta_S{}^i)$. The previous discussion results have been followed to design a fuzzy guidance algorithm in this study. The details are omitted in this paper.

5. PROPOSED VEHICLE NAVIGATION PROCESS

The proposed navigation strategies for navigation at the previously-mentioned two types of nodes are detailed in this section.

5.1 Line-following Navigation Strategy for Check Nodes

The navigation strategy of line following is adopted when the vehicle passes a check node in its navigation path. The vehicle uses mainly the fuzzy guidance technique described in the last section during the line following procedure. The details are described as follows.

Algorithm 3. *Line-following navigation*.

Step 1. Turn the vehicle toward the check node N_c to be reached.

Step 2. Calculate the distance from the current vehicle location L_v (recorded in the odometer) to the location of N_c (recorded in the learned data), and denote the distance by d_1.

Step 3. Move the vehicle forward.

Step 4. Turn accordingly if the output steering angle of the fuzzy guidance algorithm is larger than zero.

Step 5. Read the odometer to get the current vehicle location L_v' and compute how far the vehicle has moved as $d_2 = |L_v' - L_v|$.

Step 6. If the difference $|d_1 - d_2|$ is smaller than 5 centimeters, then end this navigation session; otherwise, turn the vehicle to the original direction of N_c specified in the learned data, and repeat Step 3 through Step 5.

Note that by Step 6 we can maintain the location precision of the line-following navigation; if the vehicle's trajectory has been deviated too far from its right path, then the navigation is stopped. Also, by turning the vehicle to the original direction of N_c, we can maintain the direction precision of the vehicle at each check node.

5.2 Curve-Following Navigation Strategy for Turn Nodes

The navigation strategy of curve following is adopted when the vehicle passes a turn node (called *initial* turn node) and moves toward another turn node (called *end* turn node) in its navigation path. The proposed curve following method uses the information that contains the global coordinates of the two turn nodes with their direction angles to make a smooth turn. An illustration for the proposed curve following technique is shown in Fig. 4. The main concept behind is to move the vehicle along a well-planned curve joining the two turn nodes by setting different speeds for the two wheels individually. An algorithm for the method is described as follows.

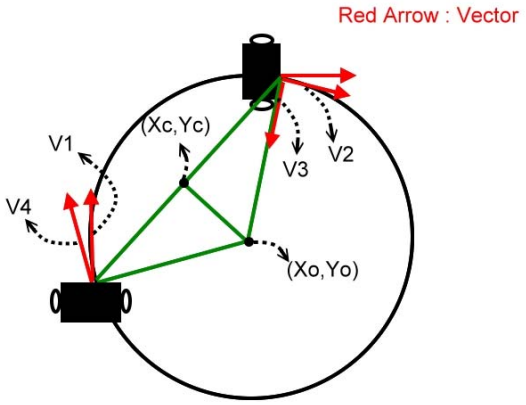

Figure 4: A figure illustrating curve following.

Algorithm 4. *Curve following*.

Step 1. When the vehicle arrives at the initial turn node denoted as N_t^1, get the global coordinates (X_1, Y_1) and the direction angle θ_1 *from the odometer*, and convert θ_1 into a vector $\vec{V}_1$ using the following equation:

$$\vec{V}_1 : \begin{pmatrix} X_1 \\ Y_1 \end{pmatrix} = \begin{pmatrix} \sin(-\theta_1) \\ \cos(-\theta_1) \end{pmatrix}.$$

Step 2. Get the global coordinates (X_2, Y_2) and the direction angle θ_2 of the end turn node N_t^2 *from the learned data*, and convert θ_2 into a vector $\vec{V}_2$ in a similar way to that for computing $\vec{V}_1$. Also, compute a vector $\vec{V}_3 = \begin{pmatrix} c \\ d \end{pmatrix}$ which is perpendicular to $\vec{V}_2$.

Step 3. Compute the middle point (X_C, Y_C) between (X_1, Y_1) and (X_2, Y_2).

Step 4. According to geometry theory, a unique circle with center (X_O, Y_O) can be obtained by giving two points and its vectors. Compute accordingly (X_O, Y_O) by the following equation:

$$X_O = X_2 + cS,\ Y_O = Y_2 + dS$$

where

$$S = \frac{(X_C - X_2)(X_2 - X_1) + (Y_C - Y_2)(Y_2 - Y_1)}{c(X_2 - X_1) + d(Y_2 - Y_1)}.$$

The details are omitted here.

Step 5. Compute a vector $\vec{V}_4$ by the following equation:

$$\vec{V}_4 = \begin{pmatrix} -\left(\dfrac{Y_O - Y_1}{X_O - X_1}\right) \\ 1 \end{pmatrix} = \begin{pmatrix} X \\ Y \end{pmatrix}.$$

Step 6. Convert $\vec{V}_4$ into an angle θ_3 by the following equation:

if $Y < 0$ and $X > 0$, then set $\theta_3 = -180 - \tan^{-1}\left(\dfrac{X}{Y}\right)$;

if $Y < 0$ and $X < 0$, then set $\theta_3 = 180 - \tan^{-1}\left(\dfrac{X}{Y}\right)$;

otherwise, set $\theta_3 = -\tan^{-1}\left(\dfrac{X}{Y}\right)$.

Step 7. Turn the vehicle leftward for the angle of θ_3 if θ_3 is larger than zero; otherwise, turn the vehicle rightward for the angle of θ_3.

Step 8. Set the speeds of two wheels individually by the following equations:

$$\frac{R_1}{R_2} = \frac{V_L}{V_R},\quad R_2 - R_1 = w$$

where V_L and V_R are the speeds of the left wheel and the right one, respectively; R_2 and R_1 are the distances from the center of the circle to the left wheel and the right wheel, respectively; and w is the width of the vehicle. An illustration of the situation and the notations is shown in Fig. 5.

Step 9. Start the curve following action until the end turn node is reached.

Step 10. Turn the vehicle to the original direction specified in the learned data, and finish the turning session.

Note that in the above curve following process, the global coordinates of the end turn node are obtained from the learned data. Therefore, even though at the initial turn node the vehicle is located at an imprecise spot due to incremental accumulations of location errors, the final vehicle location at the end turn node will become precise again. This is a merit of our method.

6. Experimental Results

Some images, which are grabbed when the vehicle navigated in an experimental trip along a learned path in an indoor environment, are shown in Fig. 6. The images also illustrate situations of line following, vehicle turning, and obstacle avoidance. Each distinct situation is marked respectively and clearly in the figure. The speed of the vehicle used in this navigation is set to be 10 cm/sec.

7. Concluding Remarks

Several techniques and strategies have been proposed and integrated into an autonomous vehicle system with simple learning and fuzzy guidance capabilities. Satisfactory navigation results have been obtained. Less data are acquired in the proposed learning process without causing instable and imprecise navigation results. In addition, a vehicle navigation method using fuzzy-control techniques has been proposed for indoor environments. Two kinds of navigation strategies, namely, line following and curve following, have been proposed to guide smoothly the vehicle in two different navigation sessions. And the experimental results show the feasibility of the proposed method.

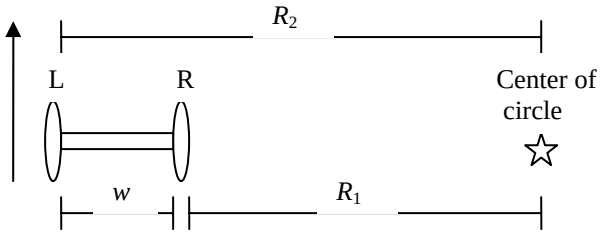

Fig. 5: An illustration of computing two wheel speeds for the case of turning rightward in curve following.

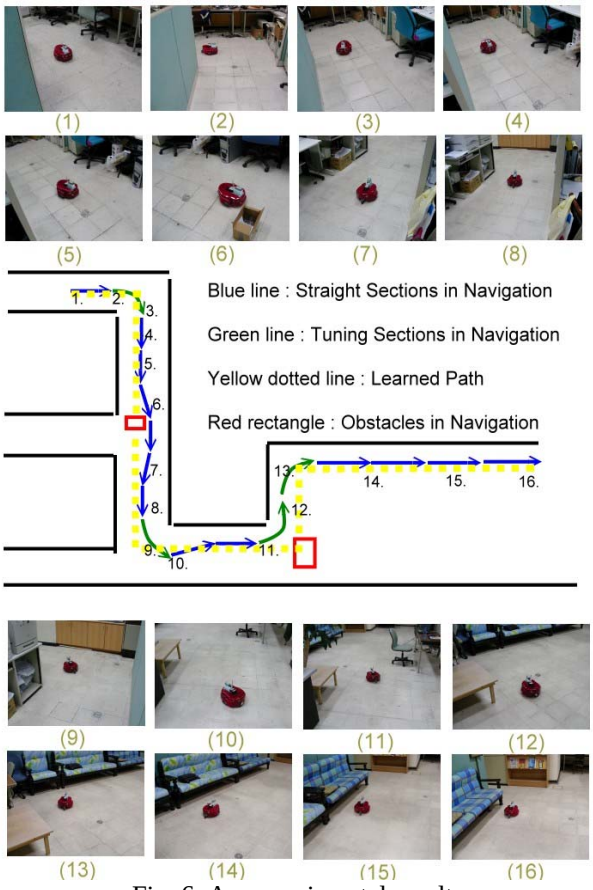

Fig. 6. An experimental result.

References

[1] Nielsen, J.; Sandini, G, "Learning mobile robot navigation: a behavior-based approach," *1994 IEEE International Conference on Systems, Men and Cybernetics*, 2-5 Oct. 1994, pp.2809 – 2814, vol. 3.

[2] Nayar, S.K.; Murase, H.; Nene, S.A., "Learning, positioning, and tracking visual appearance," *1994 IEEE International Conference on Robotics and Automation*, 8-13 May 1994, pp.3237 – 3244, vol.4.

[3] Bianco, G.; Zelinsky, A., Lehrer, M, "Visual landmark learning," *2000 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 31 Oct.-5 Nov. 2000, pp. 227 – 232, vol.1.

[4] Mata, M.; Armingo, J.M.; de la Escalera, A., Salichs, M.A., "Using learned visual landmarks for intelligent topological navigation of mobile robots," *ICRA '03. IEEE International Conference on Robotics and Automation*, 14-19 Sept. 2003, pp. 1324 – 1329, vol. 1.

[5] Gaussier, P.; Joulain, C.; Zrehen, S., Banquet, J.P.; Revel, A., "Visual navigation in an open environment without map," *Proceedings of 1997 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 7-11 Sept. 1997, pp. 545 – 550, vol. 2

[6] K. P. Li and W. H. Tsai, "Autonomous land vehicle guidance in complex room environments by computer vision techniques," *Proceedings of 1996 International Conference on Image Processing and Character Recognition* (*part of 1996 International Computer Symposium*), Kaohsiung, Taiwan, Dec. 1996, pp.9-16.

[7] Y. Tsukamoto, "An approach to fuzzy reasoning method," in M. M. Gupta, R.K. Ragade, and R. R. Yager, Eds., *Advances in Fuzzy Set Theory and Applications*, Amsterdam: North-Holland, pp. 137-149, 1979.